Using Lexicalized Grammars and Headedness for Approximate Plan Recognition

Christopher W. Geib,

University of Edinburgh
School of Informatics
Edinburgh EH8 9LW, Scotland
cgeib@inf.ed.ac.uk

Abstract

This paper presents a new algorithm for plan recognition using an action grammar formalism based on Combinatory Catagorial Grammar, that requires a significant shift in thinking about the problem of plan recognition. This approach makes significant use of the concepts of *lexicalization* and *headedness* from natural language parsing. It argues that lexicalization of action grammars can help plan recognition address a number of technical and efficiency questions, and that headedness in action grammars is previously unrecognized and an important addition to plan recognition theory in its own right.

Introduction

Some very effective prior research on Plan Recognition (PR) has been based on models of plan execution (Bui, Venkatesh, & West 2002; Avrahami-Zilberbrand & Kaminka 2005; Geib 2006). This view has effectively allowed researchers to address a number of previously open research questions, however, it also requires these algorithms to hypothesize the root goals and state of the entire collection of subplans for all viable root goals at each time point. (Geib 2004) has pointed out that for plan libraries with specific characteristics this can result in the algorithm maintaining an exponential number of goal hypotheses. For real world domains maintaining this will produce an unacceptable runtime.

To address this limitation, we describe a different approach to plan recognition and present an implementation in a system called, ELEXIR(Engine for LEXicalized Intent Recognition), that is least commitment in building the hypothesis for the observations. It is based on the natural language parsing (NLP) concept of a *lexicalized* plan library based on Combinatory Catagorial Grammars (CCG) (Steedman 2000) and the concept from NLP of *headedness* (Sag, Wasow, & Bender 2003). We are not the first work to make use of ideas from NLP in PR (See for example (Carberry 1990; Pynadath & Wellman 2000; Vilain 1991)) we are the first to apply headedness and mildly context sensitive grammars, like CCG, to this task.

To establish some basic terminology, we describe PR as taking in as input a set of *observations* and a *grammar* specifying the acceptable sets of observations. PR then *parses* these observations to produce *explanations* that organize the observations into a structured representation of the meaning of the collection.

Motivation

The vast majority of plan recognition systems are tested on plan libraries with a very small set of root plans, and in general work quite well on restricted sets of roots. However since for algorithms that must hypothesize the root goals and the state of the agent's subplans, the algorithm's runtime most often scales with the number of roots in the plan library, a fact that is deemphasized by testing on a small set of root goals.

This problem is made worse if there are a relatively small set of observable actions. If the number of actions is small relative to the number of root goals (a common feature of human behavior domains) we are pushed in to a space where individual actions are not highly informative of the root goals. It is rather sets of observations, and even sets of subgoals and subplans that are informative of the agent's goals. This means such an algorithm must maintain a larger set of hypothesized root goal and plan structures since the observed actions do not differentiate between them.

Consider for example the domain of computer network security, the prerequisite of almost any hostile activity is reconnaissance. Now while observing scanning and probing actions on a network is highly correlated with hostile activity it does not predict the kind of activity. Is the attacker interested in stealing data, executing a denial of service, or just installing Netcat to allow him to "harmlessly" route traffic through the attacked machine. In exactly this case, (Geib 2004) has shown that plan libraries with specific kinds of structure produce an exponential number of possible explanations.

The space of possible root goal hypotheses grows still further when we take seriously the possibility of multiple root goals. In real human domains, multiple concurrent and interleaved plans are common. Consider a "stay at home parent" that is doing the laundry, cleaning the house, and watching the kids all at the same time. No single goal will explain all of their activities. We must see this as three concurrent and interleaved goals.

In the end, efficient plan recognition simply can't afford to hypothesize all of the possible root goal sets that could be leading to the observed actions. Rather than building the

set of all hypothesis that could explain the observed actions, we propose a least commitment approach, that builds only that part of the explanation that is required by each of the actual observations. Such an approach presents two critical questions.

1. How much of the plan structure should be built? Traditional plan libraries for PR based on HTN like plan structures (Ghallab, Nau, & Traverso 2004) do not provide us with any hints as to how much of the plan should be built. We will turn to lexicalized grammars and the concept of headedness from NLP to answer this question.
2. How to estimate the probabilities of the root goals given least commitment explanations? We will follow prior work in considering probabilistic PR as a problem of weighted model counting (Geib 2006). However this does not address the question of how the probabilities for each of the models will be computed.

In the rest of this paper we will address these questions in turn, and present ELEXIR, a probabilistic plan recognition system, based on these ideas.

Lexicalized Building of Plan Hypothesis

In this work, we will argue for building plan structures based on a particular lexicalized grammar formalism. This is not intended to be a complete introduction to formal grammars. We refer the interested reader to (Aho & Ullman 1992) for a background on formal language theory and to (Steedman 2000) for background on CCGs. This section is intended as a brief introduction to lexicalization of grammars and CCGs for those already familiar with traditional formal grammars.

Traditional grammars like context free and context sensitive grammars have two parts: 1) A set of rules that map a sequence of non terminal symbols to a sequence of terminal and non-terminal terminal symbols, and 2) a lexicon that maps each possible observation to (possibly a sequence of) terminal symbols. Note that in these grammars partial ordering on the right hand side of the grammar rules is handled by creating multiple rules. This will differ markedly with our later grammatical formalism.

It is well known that, limits in the form of the grammar rules create different expressive classes. For example if the left hand side of the grammar rules is limited to a single non-terminal then the grammar is known to be context free. Context sensitive grammars allow a much less restrictive set of grammatical rules.

The objective behind lexicalization is to remove explicit language specific grammar rules by moving the information they contain into the lexicon. Traditional context free and context sensitive grammars have a *large language specific* set of rules and a lexicon pairing terminals with a *small largely non-language specific* set of pre-terminals. Lexicalized grammars have a *small universal set* of syntactic productions or combinators and a *large language specific lexicon* pairing terminals with a *large language specific set of pre-terminal types*.

Lexicalization of the grammars has allowed the exploration of more fine grained formal language categories. It has resulted in the identification of the class of Mildly Context Sensitive Grammars (MCSG) that fall between CFGs and CSGs in complexity. (Geib & Steedman 2007) have argued for the use of MCSGs for PR on the basis of complexity while previous work on PR as parsing has focused on probabilistic and non-probabilistic CFGs or CSGs.

In this work we demonstrate the use of a particular MCSG namely Combinatory Catagorial Grammars (CCG). The interested reader is referred to (Steedman 2000) for a more complete treatment of CCG in an NLP context. In the following section we define CCGs and show how they can be used to encode plan grammars.

Consider the simple abstract hierarchical plan shown in Figure 1. In this plan to achieve goal state **G** the agent must

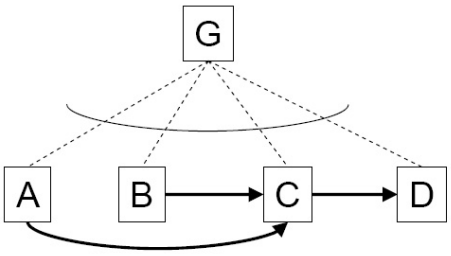

Figure 1: An abstract plan with partial order causal structure

perform actions **A**, **B**, **C**, and **D**, and further **A** and **B** must be executed before **C** but are unordered with respect to each other. This structure could be represented as an Hierarchical Task Network (HTN) (Ghallab, Nau, & Traverso 2004) as:

$$(m1, \quad G, \quad \{A, B, C, D\}, \quad \{(1 \prec 3), (2 \prec 3), (3 \prec 4)\})$$

We will now present more formal definitions for CCGs and then show how they can be used to encode this method.

CCG Definitions of Plans

To define a plan library with CCGs each observable action is associated with a set of categories that encode the action grammar. We define the set of possible categories recursively as:

Atomic categories : The plan library is assumed to have a finite set of basic action categories C. (i.e. $A, B, ... \in C$.) These categories correspond to propositions that describe world states that result from executing actions. For example, being in the bank or grasping the gun.

Complex categories : $\forall Z \in C$ and a non empty set $\{W, X, ...\} \subset C$ then $Z \backslash \{W, X, ...\} \in C$.

The complex category $Z \backslash \{W, X, ...\}$ is a functor category that takes an unordered set of arguments $\{W, X, ...\}$ and produces a result Z. For actions in a planning domain, the semantics of the $\backslash$ is basically that of subgoal that precedes. Thus an action with the category $A \backslash B$ is a function that results in A in contexts where B is already true. Traditional CCGs also provide a forward slash that allows for building functor categories where the arguments are to be found after the observed action. Such rightward categories are the key to

incrementality in NLP and we believe they provide a similar functionality in PR contexts however to simplify our exposition we will limit our discussion here to leftward slashed categories.

The particular formulation of CCG we have described here is closely related to that of (Hoffman 1995; Baldridge 2002) in allowing multiple unordered arguments. This will be critical for application to PR contexts with partially ordered plans.

To see how these categories work in practice consider encoding our example method. We will use lower case letters to denote observations of the given category. Thus, we can create the following lexicon to assign possible observations to action categories:

$$a \vdash A,\ b \vdash B,\ c \vdash G'\backslash\{A,B\},\ d \vdash G\backslash G'$$

Where $a \vdash A$ indicates that an observation of type a is of category A. Note that all of the structure associated with the HTN is now included in the functor categories for observations c and d. We should note that while none of the categories shown here cover more than one level of an HTN definition there is nothing that limits them in this way.

To combine a stream of observations into a more complex structure we use *combinators* to combine the categories of the individual observations into higher level structures. For the simple examples in this paper we will limit ourselves to function application and composition to combine categories:

$$\begin{array}{ll} \textit{application:} & Y\ \ X\backslash Y\ \ \Rightarrow\ \ X \\ \textit{composition:} & X\backslash Y\ \ Y\backslash Z\ \ \Rightarrow X\backslash Z \end{array}$$

A number of other combinatory rules are possible and sometimes used in CCGs, we refer the interested reader to (Steedman 2000) for details, and we will leave the exploration of these combinators in PR context for future work.

To see how these combinators combine with the lexicon to allow for parsing observations to recognize high level goals consider the following derivation that recognizes the sequence of observations: a, b, c.

$$\dfrac{\dfrac{\begin{array}{ccc} a & b & c \\ \dot{A} & \dot{B} & \overline{G'\backslash\{A,B\}} \end{array}}{G'\backslash\{A\}}<}{G'}<$$

Figure 2: Building an Explanation by Parsing Observations with CCGs

We now need to more clearly define what we mean by an *explanation* for a set of observations. Since we are committed to having multiple possible concurrent goals, for the purposes of this discussion we will describe an explanation as a sequence of CCG categories. Thus in the previous example { G' } would be one resulting explanation.

While the above parse shows how CCGs can be used to build explanations for observed actions, we have carefully avoided the question of why the action lexicon should be the one we have provided. In fact, there are other possible assignments of actions to categories that would produce an equivalent parse for these actions and this plan structure. For example the following is a second lexicons that can produce the same parse for this set of observations.

$$a \vdash A,\ b \vdash B,\ c \vdash C,\ d \vdash (G\backslash C)\backslash\{A,B\}$$

In the different lexicons the categories that provide a plan's gross structure can be associated with different actions. For example, in our original lexicon the categories for c and d provide the structure of the plan, but in the latest lexicon all of the structure is confined to the category for d. These lexicons will result in exactly the same parsed results for complete sequences that generate a G, however they do differ in the intermediate structures that are built.

Representational choices about which categories to assign to which observations must be made at the action grammar's design time. Thus, if we are going to use CCGs to represent our plan grammars, we need to examine this new modeling consideration.

Similar modeling issues do occur in HTN/CFG representations of action hierarchies in the form of choosing the subgoal decomposition. With their long tradition in planning and reasoning, intuitions about what is and isn't a subgoal seem natural in a planning domain. However categories have a multidimensional aspect to them (they can function effectively as a tree spine rather than the decomposition of a single level). This makes a traditional hierarchical interpretation of them less intuitive. To address some of these and other issues linguists have invented the concept of *headedness* (Sag, Wasow, & Bender 2003).

Headedness in Plans

Consider the case where we see Leslie threaten a bank teller with a gun. While there are a great many actions that have to come both before and after this action for Leslie to successfully complete a plan for robbing the bank, this act is central to the plan. The plan cannot be performed without intimidating the teller. In this sense, this action of threatening the teller is the *head* of Leslie's plan to rob the bank.

Next consider the case of Leslie walking into a retail store (without the gun). We can again effectively infer that Leslie is shopping, even if we have seen no acts that went before this, and even if Leslie doesn't, carefully consider any merchandise in the store or actually purchase anything. Again the action of entering the store is the head of the plan for shopping.

Headedness in NLP grammars has long played a crucial role, and explains why we talk of noun phrases, verb phrases and the like. It is because the noun or verb is the head of these respective phrases. While NLP research has had a great deal to say about headedness in natural language grammars, our purpose here is to argue for its realization in PR domains.

In the case of lexicalized action grammars, taking headedness seriously reduces to the identification of a single observable action that can be assigned a category with the structure that results in the root of the plan or sub-plan under consideration. This corresponds to making the design decision discussed in the previous section: determining which action will be assigned the category that generates the root

category of the plan or sub-plan. Given the many uses we think of actions as having, making a claim about the plausibility of headedness in action plans may seem unusual. However, there are a number of arguments in its favor.

Since any given observation can have multiple categories assigned to it, we are not claiming that an observation can only ever play the single role of being the head of a single plan. Actions are less constrained in the roles they can play within their grammars than words, and therefore, we would expect that the same action could have a large number of possible categories assigned to it in the plan lexicon.

It should be clear that, as in NLP, we assume that headedness applies not only at the root goal but also for subgoals as well. For example, Leslie must acquire a gun to rob the bank, and entering a gun store would be the head of the sub plan of shopping for the gun. Thus while a particular observed action must be associated with the root goal it is not the case that it would also be carrying the weight of determining the whole of the structure for the plan, any more than making a choice about how to decompose one subgoal determines the decomposition for the rest of an HTN plan.

Further, actions that participate in a very small number of possible plans (i.e. taking a gun into a bank) do seem to provide us with natural instances of headedness in plans arguing for its naturalness. This also seems to correspond to intuitions that people sometimes plan the head action of the plan and then complete their plans both forward and backwards from this central action. (i.e. Consider the case of planning an evening out for dinner. Choosing the restaurant and the time is certainly done before deciding on when and where pre-dinner drinks might be or where one would go after the dinner.)

However such naturalistic arguments would only be weak evidence if not for a final much stronger argument that, headedness is not only required for the use of the formalism, but careful selection of plan heads is necessary to capture partial ordering in plans and can provide a powerful method of controlling the space of possible explanations to be considered in PR. If we adopt the approach discussed so far, headedness allows us to define exactly how much of the plan hypothesis space should be built on the basis of the observed actions, without recourse to arbitrary limits or parameters. Instead we implement the following principle in building plan structures:

Principle of minimal head justified explanation: In building explanations we never hypothesize any plan structure beyond that provided by the categories of the observed actions.

This principle provides a clear answer to the first question posed in the introduction concerning how much plan structure to build. Further it limits the size and scope of the plan hypothesis space for PR. It takes the idea of building the minimal structures that are consistent with the observed actions very seriously and as we will see in the next section results in a very simple algorithm for generating explanations of a set of observed actions.

Procedure *BuildExplanations*($\sigma_1 ... \sigma_n$) {
  $ES = \{E_0\}$;
  For $i = 1$ to n
    $ES' = \emptyset$;
    For each $exp \in ES$
      For each $c \in C_{\sigma_i}$;
        $exp' = exp \cup c$
        $ES' = ES' \cup exp'$
        Compute the set of all explanations
          $ES_{close(exp',c)}$ that can result from the
          application of one of the combinators
          to any two categories in exp' preserving
          the ordering of the observations.
        $ES' = ES' \cup ES_{close(exp',c)}$;
      End-for;
    End-for;
  $ES = ES'$;
  End-for;
return ES; }

Figure 3: High level algorithm for explanation generation.

Building Explanations in ELEXIR

Effectively ELEXIR produces explanations from a set of action observations by assigning to each observation each of its possible categories and then to applying the applicable combinators to produce the complete set of possible explanations for the observations. Effectively this is also exactly the same process carried out in NL parsing. However there are a few differences.

For example, we can't assume that we know a-priori how many observations there will be, or that we can bound this number. Further, we do not assume that all of the observations must contribute to a single goal, and we cannot assume that we have seen all of the observations associated with the plan. Many well known parsing algorithms like CKY, even when modified for CCGs (Steedman 2000), leverage some or all of these assumptions in their processing and as such are unacceptable for our purposes.

Further, some probabilistic NL parsing algorithms do not produce the complete set of parses. They only consider assigning the highest probability categories to each observation. While in principle we have no objection to this, it is an area for future work to determine if this would result in a significant loss of accuracy in a PR system. This short cut is not implemented here.

Instead, we have a simple algorithm captured in the pseudo code given in Figure 3. In effect this algorithm does nothing more than create the set of all explanations that result from the incremental closure of the categories introduced by the current observation and the existing categories.[1] For example, given our original lexicon and the observations: a, b, c, d the algorithm produces the following

[1]To date we have not explicitly addressed the problem of *spurious ambiguity* in CCGs (Steedman 2000), however we believe that to the degree this presents a problem for PR, the reductions to normal form parsing (Eisner 1996) used to address this problem in NLP will work in PR as well.

explanations: (1) {*G*}, (2) { *G'*, *G* \ *G'* }, (3) { *A*, *G'* \ {*A*}, *G* \ *G'* }, (4) {*B*, *G'* \ {*B*}, *G* \ *G'*}, (5) {*A*, *B*, *G'* \ {*A*, *B*}, *G* \ *G'*}.

Note that in each case the ordering of the actions has been preserved. This is necessary to correctly enforce the directionality information encoded in the category slashes. While preserving the ordering constraints, our implementation of this algorithm does violate the "adjacency restriction" that requires that only directly adjacent categories be able to be combined by a combinator that is usually enforced in CCG grammars. While this can in theory increase the computational complexity of the parsing algorithm it is necessary to handle multiple interleaved plans.

It is worth discussing why the algorithm should produce explanations 2-5 in the above example. They are included to account for the case where any of the categories will be used in some other, as yet unseen, plan. Under the assumption that a given category can only contribute to a single plan, if these categories are consumed at the earliest opportunity they will be unavailable for later use. Therefore, in the interest of these future possibilities we must produce these explanations.

Thus, we have outlined an algorithm for computing a set of explanations that make the minimal commitments required by the observed actions, thus answering the first of the questions we raised in the motivation section. We now move to address the question is now how to use this set of explanations to determine the relative likelihood of various specific goals.

Computing Probabilities in ELEXIR

Traditionally in probabilistic plan recognition the objective is to compute the conditional probability for all of the possible root goals (Charniak & Goldman 1993). As we have already argued, the space of possible explanations is very large and can in some contexts be infinite. Further it is not clear that the space of explanations for observed actions has the kind of properties that are argued for in NLP contexts that make approximating a probability distribution over such a space possible.(Booth & Thompson 1973). However, the explanations that we have generated in the last section are more amenable to a different approach: weighted model counting.

Weighted model counting in plan recognition works in the following manner. Assuming we can compute the exclusive and exhaustive set of explanations for a given set of observations, and that we can compute the conditional probability of each explanation given the observations, then the conditional probability for any given goal is given by the following formula:

Definition 1.1

$$P(goal|obs) = \sum_{\{exp_i|goal \in exp_i\}} P(exp_i|obs)$$

where $P(exp_i|obs)$ is the conditional probability of explanation exp_i, and the conditional probability for the goal is just the sum of the probability mass associated with those explanations that contain the goal of interest.

But note this approach critically relies on 1) an exclusive and exhaustive set of explanations for the observations and 2) being able to compute the conditional probability for each explanation. While in the next subsection we will show how to compute the probability for an explanation, there is a conflict between computing only minimal head justified explanations and computing the complete and covering set of explanations. To see this we return to our example of Leslie robbing the bank. Consider the following plan grammar:

$$graspBag \vdash GB, \quad graspGun \vdash GG,$$
$$enterBank \vdash IB, \quad graspCash \vdash GC,$$
$$threaten \vdash (ROBBANK \backslash IB) \backslash \{GB, GG\},$$
$$handOverGun \vdash (SELLGUN \backslash IB) \backslash \{GB, GG\}$$

and the following series of observations:

$$(graspBag, t0), (graspGun, t1), (enterBank, t2)$$

There is only one minimal head justified explanation for these observations and that is that they are all being engaged in for their own ends. Since none of the observations is of a category that allows for the combination with the other categories, the only consistent explanation is that each of these actions as being done as a separate root goal. This highlights a significant problem for the principle of minimal head justified explanations. They are unable to talk meaningfully about a plan until the head of the plan has been observed. This clashes quite seriously with our intuition that all three of the observed actions could be forming part of a larger plan.

To address this limitation. we will describe two different mechanisms for understanding the likelihood of different root goals. Both will be based on weighted models counting but the first will be a bottom up process constrained by our minimal head justified explanations, and the second will address the need to look at goals who's head has not yet been observed. Both of them will require computing the conditional probability of each explanation.

Computing the Probability of an Explanation

While there are a number of different probability models used for CCG parses in the NLP literature (Hockenmaier 2003; Clark & Curran 2004) for this work we will use a particularly simple model, extending one described in (Hockenmaier 2003). For an explanation, *exp*, of a sequence of observations, $\sigma_1 ... \sigma_n$, that results in m categories in the explanation, we define the probability of the explanation as:

Definition 1.2

$$P(exp|\{\sigma_1 ... \sigma_n\}) = \prod_{i=1}^{n} P(cinit_i|\sigma_i) \prod_{j=1}^{m} P(root(c_j))K$$

Where $cinit_i$ represents the initial category assigned in this explanation to observation σ_i and $root(c_j)$ represents the atomic head result category of the jth category in the explanation.

The first product represents the probability of the given observations having the specified CCG categories. This is standard in NLP parsing and assumes the presence of a probability distribution over the possible categories that a given observation can be mapped to. In NLP such probabilities are

usually learned using large corpora of parsed text (Clark & Curran 2004).

The second product (and its associated constant) is designed to capture the probability that, each category in the explanation will not be part of a larger plan but instead represents a separate rooted plan instance. Unlike the first term, this is not part of traditional NLP models for two reasons. First, in NLP it makes little sense to consider the probability that a speaker would utter a collection of nouns and verbs each as an isolated unrelated declarative statement. Second, in most NLP contexts the sequence of observations is further known to be a complete sentence (usually parsed text contains punctuation marks indicating sentence boundaries). Again in this setting it makes little sense to consider the probability that the sequence is anything but a single complete sentence. However these assumptions do not in general hold in the PR domain. It is more than possible for a given sequence of observations to contain multiple plans of varying lengths, or to only cover fragments of multiple plans being executed (consider a set of multi-day plans).

To address recognition in these domains we take a slightly extreme position within the PR community and assume that any action could be done for its own sake. This is different than much work on PR. This work assumes that, however unlikely, it is completely acceptable for any given action to be a root goal and to be executed by itself without regard to a more complex goal. Therefore, we must assume a prior probability for each atomic category as a root goal. We believe it is relatively straightforward to collect frequencies from real world data to provide such priors.

Given this set of assumptions, to understand the derivation of the second term in the above definition, we denote the set of all values of $root(c_j)$ for a given explanation, exp as $goals_{exp}$ and the probability of this particular set of categories being adopted as root goals as $P(goals_{exp})$. We represent the probability of an agent adopting a category c as a root goal as $P(c)$ with each goal instance being chosen (or rejected) independently.

Since in ELEXIR we want to allow for multiple instances of a given category in $goals$ (it is acceptable for $root(c_i) = root(c_j)$ where $i \neq j$), we assume that individual goals are sampled as a geometric distribution. Therefore, $P(c)$ represents the probability that category c is added as a root goal in the explanation and we keep sampling to see if there are more root instances of c. This allows us to write the probability that there will be exactly n root instances of any category c as $P(c)^n(1 - P(c))$.

This is almost certainly incorrect — intuitively the probability of multiple instances of a single goal decreases far more rapidly than this, making this an over estimate of the likelihood of the goals. However, in practice this simplification seems benign, and adding a more complex model has not yet been necessary. The theory will support more more sophisticated models, and we see examining these as an area for future research.

Assuming $|goals_c|$ represents the number of root instances of category c in the explanation allows us to define:

$$P(goals) = \prod_{c \in goals} P(c)^{|goals_c|}(1 - P(c)) \prod_{c \notin goals} (1 - P(c)).$$

Moving all of the $1 - P(c)$ terms into a single product allows us to write this as:

$$P(goals) = \prod_{c \in goals} P(c)^{|goals_c|} \prod_{\forall c \in C} (1 - P(c))$$

Now note, the second term is a product over the set of all atomic categories, C, in the plan library, and is actually a constant across all explanations:

$$P(goals) = \prod_{c \in goals} P(c)^{|goals_c|} K$$

Rewriting this in terms of the instances in the explanation finally yields the term seen in Definition 1.2.

$$P(goals) = \prod_{j=1}^{m} P(root(c_j) K$$

Computing Goal Probabilities Bottom-up

Having defined an algorithm that builds the set of possible explanations, and defined a method of computing the probability for each such explanation, we are now in a position to compute the conditional probability of each of the root goals that occurs in any of the explanations. Following Definition 1.1 it is straight forward to loop over the set of explanations computing the conditional probability for each root goal that occurs in any explanation of the observations.

However, as we have discussed, given the principle of minimal head justified explanations, this bottom up process has a critical limitation. It assumes that there will be no more observations. It has not considered any explanations that could result from the observed actions contributing to a goal for which the head has not yet been observed.

In some cases this will be perfectly acceptable. If in, our bank robbing example we knew for a fact that Leslie was not going to perform any more actions then we would have to assume that the grasping the bag and gun and entering the bank were in fact unrelated actions. However, this is unsatisfying for most real world situations where we know more observations are possible. To address this need, in the next subsection we outline a top-down, goal directed metric of how much the observed actions could contribute to a particular goal of interest.

Computing Top-down Probabilities

We have already argued that the space of possible explanations, given the availability of multiple goals and large numbers of observations, is too large to be effectively searched. Therefore, our goal is not to compute a probability distribution over the space of all possible permutations of root goals but instead to produce a metric that will indicate how likely it is that a particular goal is being executed given the observed actions.

To inform this, reconsider the Bank robbing example. The reason that it seems likely that Leslie is going to rob the bank is because all three of the observed actions could be arguments to a category that has robbing the bank as its head. However, all three of the observed actions could also be arguments to the final category in our previous lexicon,

namely selling the gun as its head. However we dismiss selling the gun to someone in the bank as being very unlikely. This seems to identify two features that should be critical in our metric: 1) the number of observed actions' categories that can act as arguments or sub-arguments to a category that has the desired goals as its head result, and 2) the prior probability of the desired goal.

Imagine in our example, hypothesizing an instance of the robbing bank goal, and adding it to our earlier explanation. The process that we have already discussed for computing explanations from observations would combine all three of the observations with the robbing bank category to produce a reduced explanation with *ROBBANK* as its sole category. This explanation has a much higher conditional probability. The root probability terms for each of the three categories have been removed and replaced with the prior probability for robbing the bank (which while unlikely in itself is much more likely then the three unrelated events happening). This suggests leveraging our existing algorithm, hypothesizing the desired goal and making use the change in the conditional probability of the explanations ($\triangle CP$) to estimate how likely it is that the agent is actually pursuing the desired goal.

Large positive shifts in $\triangle CP$ indicate two things. First, there are observed actions with categories that can play the role of arguments to a category with the desired head, (root probability terms for these categories drop out of the conditional probability for the explanation.) Second, this rise in probability from the loss from the categories that contribute to the desired goal as roots overshadows the cost associated with the introduction of the root goal.

Negative shifts in $\triangle CP$ indicate the opposite. Either there weren't sufficient gains from actions that contribute, or the cost of the desired goal was too high.

It is important to note that since the observed actions may be able to act as arguments to a sub-category of a category of interest the process of identifying which observations could be playing a role in the desired goal is slightly more complex than simply extending the explanations with a hypothesized new category. Instead we will compute the *maximal cancellation* for the explanation given the category. We will return to discuss this in more detail but first we outline the specific metrics to be used.

ELEXIR is computes two statistics based on $\triangle CP$.

Maximum $\triangle CP$ ($\mathbf{max}\triangle CP$): For each explanation of the observations loop over the set of all categories that have the desired head result compute the $\triangle CP$ for the maximal cancellation for each explanation and category pair. Store the maximum across all explanations.

Average maximum $\triangle CP$ ($\widehat{\mathbf{max}}\triangle CP$): Average the $\mathbf{max}\triangle CP$ values computed for each explanation.

Keep in mind that both of these statistics may be negative. A given goal of interest my actually make the explanations less likely. The algorithm for computing these statistics is shown in figure Figure 4.

To make this algorithm more precise, we define the *maximal cancellation* of a new category against an existing explanation as the resulting explanation when the maximal number of categories in the original explanation that can fulfill an argument role for the category that achieves the desired goal have been removed. The largest $\triangle CP$ gains must be achieved when the maximal number of existing categories fulfill roles as arguments in the desired goal category (since their root goal probability term will be removed from the explanations CP.)

```
Procedure ComputeDCP(goal, expSet) {
    For each e in expSet
        For each c in the lexicon st. goal = head − result(c)
            Add to e the category c
            Compute the maximal cancellation for
                e by removing from e all root
                categories that could contribute to c
            Compute △CP;
            End-for;
        Store max△CP for each e
        End-for;
    Compute max̂△CP
    Compute max△CP}
```

Figure 4: Algorithm for $\mathbf{max}\triangle CP$ and $\widehat{\mathbf{max}}\triangle CP$.

As we have already alluded producing the maximal cancellation must be sensitive to the possibility that categories may act as an argument to a subcategory of the desired goal. In more traditional planning parlance it may contribute to a subgoal. Therefore the process of computing the maximal cancellation must consider all of the possible completions for the goal category's arguments. Like traditional hierarchical planning, this can require decomposing a subgoal. In this process this corresponds to expanding one argument by replacing it with a functor category from the lexicon that results in the appropriate category. Consider the explanation $\{A, B, C\}$ looking for a D where the plan grammar includes $(D\backslash P)\backslash A$ and $P\backslash\{B, C\}$. In this case, all of the previously observed actions could contribute to achieving goal D however this can only be seen if atomic category P is expanded.

Allowing this kind of category decomposition also requires that all possible permutations of unordered arguments be considered. Consider again the simple explanation $\{A, B, C\}$ in the context where we are looking for D but this time the grammar includes $D\backslash\{P, A\}$ and $(P\backslash B)\backslash A$. If we first remove the A argument to D and then consider decomposing P we get $\{B, C\}$ as the maximal cancellation, but in the other order we can get $\{C\}$ as the maximal cancellation because the A is still in the explanation.[2] Thus computing the maximal cancellation for an explanation and a category does clearly imply that we have to look at all of the possible cancellations and orderings for the categories.

Complexity

Given that this process must be performed pairing every explanation with every category that has the desired head result this seems as though it might be computationally expensive.

[2]We are looking at overloading category cancellation as an area for future work. This would obviate the need for considering multiple orderings in this case.

We take some solace however in the realization that, any complete algorithm will be required to consider this same space. Further, algorithms that consider the case of multiple possible root goals and initially hypothesize the goal set are required to consider all of these possible plans interleaved with all of the other possible hypothesized goals, a far worse search space.

Implementation of ELEXIR

The approach and algorithms described here have been implemented in Common LISP and successfully tested on a number of small example problems including the bank robbery example discussed here and a hand crafted selection of test problems. While initial tests show impressive performance our data for claims in this area is still anecdotal. Given the significant differences between this approach to PR and much of the existing work we are currently in the process of assembling data from a large real world test domain in order to do a more thorough evaluation of the algorithm itself and to compare it to other systems.

Conclusions

This paper has covered a considerable amount of ground both in theoretical concerns and applied issues. The main conclusions can be summarized as:

- We argue, the size of the plan hypothesis space for plan recognition in real world domains is too large to admit efficient computation of exact probabilistic solutions.
- We describe an algorithm that uses headed, lexicalized, mildly context sensitive grammars (specifically CCGs) in a least commitment approach to building plan hypothesis to avoid the size of the search space.
- We defined two metrics based on the conditional probability of explanations for the observations that can be used to compare alternative hypothesis about the agent's goals.

Acknowledgments

The work described in this paper was conducted within the EU Cognitive Systems project PACO-PLUS (FP6-2004-IST-4-027657) funded by the European Commission.

References

Aho, A. V., and Ullman, J. D. 1992. *Foundations of Computer Science*. New York, NY: W.H. Freeman/Computer Science Press.

Avrahami-Zilberbrand, D., and Kaminka, G. A. 2005. Fast and complete symbolic plan recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Baldridge, J. 2002. *Lexically Specified Derivational Control in Combinatory Categorial Grammar*. Ph.D. Dissertation, University of Edinburgh.

Booth, T., and Thompson, R. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers* C-22(5):442–450.

Bui, H. H.; Venkatesh, S.; and West, G. 2002. Policy recognition in the Abstract Hidden Markov Model. *Journal of Artificial Intelligence Research* 17:451–499.

Carberry, S. 1990. *Plan Recognition in Natural Language Dialogue*. ACL-MIT Press Series in Natural Language Processing. MIT Press.

Charniak, E., and Goldman, R. P. 1993. A bayesian model of plan recognition. *Artificial Intelligence* 64(1):53–79.

Clark, S., and Curran, J. 2004. Parsing the wsj using ccg and log-linear models. In *ACL '04: Proceedings of the 42th Annual Meeting on Association for Computational Linguistics*, 104–111. Morristown, NJ, USA: Association for Computational Linguistics.

Eisner, J. 1996. Efficient normal-form parsing for combinatory categorial grammar. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, 79–86.

Geib, C., and Steedman, M. 2007. On natural language processing and plan recognition. In *Proceedings of IJCAI 2007*.

Geib, C. 2004. Assessing the complexity of plan recognition. In *Proceedings of AAAI-2004*, 507–512.

Geib, C. 2006. Plan recognition. In Kott, A., and McEneaney, W., eds., *Adversarial Reasoning*. Chapman and Hall/CRC. 77–100.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory and Practice*.

Hockenmaier, J. 2003. *Data and Models for Statistical Parsing with Combinatory Catagorial Grammar*. Ph.D. Dissertation, University of Edinburgh.

Hoffman, B. 1995. Integrating 'free' word order syntax and information structure. In *Proceedings of the 1995 Conference of the European Chapter of Association for Computational Linguistics*, 245–252.

Pynadath, D., and Wellman, M. 2000. Probabilistic state-dependent grammars for plan recognition. 507–514.

Sag, I.; Wasow, T.; and Bender, E. 2003. *Syntatic Theory*. CSLI Publications.

Steedman, M. 2000. *The Syntatic Process*. MIT Press.

Vilain, M. 1991. Deduction as parsing. In *Proceedings of the Conference of the American Association of Artifical Intelligence (1991)*, 464–470.